

# A Study of User Controls in Generation of Engraved Portraits

Karthik Ramanathan<sup>+</sup>, Fred Martin<sup>++</sup>

## Abstract

This paper deals with the conversion of a digital image of a person into a portrait engraved on the surface of a material (white foam with a dark surface). In this approach, a series of holes of varying diameter are drilled into the surface. A mapping is made from a set of numbers (RGB values) in the input image to a set of diameters for the holes to be drilled. The holes expose a white inner core on a dark surface. By varying the diameter of each hole in correspondence with the RGB-values, a portrait of the image is produced.

The algorithm is implemented in a Java application and a set of “user controls” are produced for controlling various parameters in the algorithm. The program is tested for different variations on the basic algorithm and on a number of test inputs (images of different people).

## Introduction

The goal is to produce a portrait of a person’s image using the Roland EGX-300 Desktop Engraver as a fabrication device. This idea comes from a motivation to have an alternate way of representing images. The hope is also to extend the use of prototyping tools like Roland EGX-300. Prototyping tools have been used for various applications like CAD and circuit design. A portrait generation would be an aesthetic extension of the same.

Portraits have been drawn and generated mainly by hand. Kiosk-drawings and professional portraits of people rely on a variety of materials like pencil, charcoal, oil paints etc. Sculptures have been around for ages. Here, the image is carved or engraved onto a physical material like wood or rock. In a sense, the image data is represented by “engraving” on the surface. The portrait engraving in this project captures this idea of “engraving” image data values onto the surface. The approach was to have a reasonably fast algorithm to produce and fabricate the portrait of a person.

This is similar to half-toning algorithms though not the same. Halftoning algorithms, generate “binary” dot patterns from a image plane(s). The term “binary” here refers to the presence or absence of dots. The transformation is from each intensity value to grains of dots spread over a region. The greater the number of dots in an output region, darker the region appears. Thus, more dots in the output means a lesser intensity value in the input image. This inversion in the transformation occurs because each “dot” is black and the background is assumed to be white. So, more the intensity value, brighter should be the output and hence, lesser the number of black dots.

---

<sup>+</sup> Department of Computer Science, University of Massachusetts Lowell. Email: [kramanat@cs.uml.edu](mailto:kramanat@cs.uml.edu)

<sup>++</sup> Department of Computer Science, University of Massachusetts Lowell. Email: [fredm@cs.uml.edu](mailto:fredm@cs.uml.edu)

In our algorithm, however, each intensity value is mapped onto a depth value of a drill hit. The depth is directly proportional to the diameter of the hole drilled because a conical tool is used [Fig. 1]. A greater input intensity however, means a greater depth (as opposed to lesser number of dots in half toning). This is because a drill hit exposes a white inner core on a black surface (as opposed to black dots on a white surface). This means that we have to perform a direct mapping from intensity values to depth of holes drilled.

### The Basic Algorithm

The input is assumed to be a 300x300-BMP file. An input image of a different size is rescaled to this dimension. The input can be described as a set of pairs,  $\{ \langle P_i, RGB_i \rangle \}$ , where

$$\begin{aligned}
 P_i &\leftarrow \text{A point on the image } (x_i, y_i). \\
 RGB_i &\leftarrow \text{The RGB value at the point } P_i. \\
 0 \leq i &< |Pixels|
 \end{aligned}$$

The algorithm is a mapping from this set to the pair  $\{ \langle P_j, Depth_j \rangle \}$ , where  $Depth_j$  represents the depth to drilled at the point  $P_j$ .

$$0 \leq j < |Drill - hits|$$

The generated portrait is a [60 x 60] matrix of drill-hits. The number of drill-hits is thus not the same as the number of pixels in the input image. The output points are arrived at by local averaging the input image over a [5 x 5] region. Thus a [300 x 300] input is mapped onto a [60 x 60] output.

### The working

Assume we are working with only one of the spectral planes (In the actual implementation, the three spectral planes are averaged to produce a single plane). The value at each point is to be mapped onto a particular diameter of the hole to be drilled. We use a conical drill. Thus, greater the depth of the hole drilled, the greater the diameter of the hole. These two quantities are linearly related (Fig. 1).

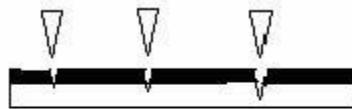


Fig. 1. Drills of different depth. The important thing to note is the triangular profile of the tool and the amount of area exposed.

### User Control

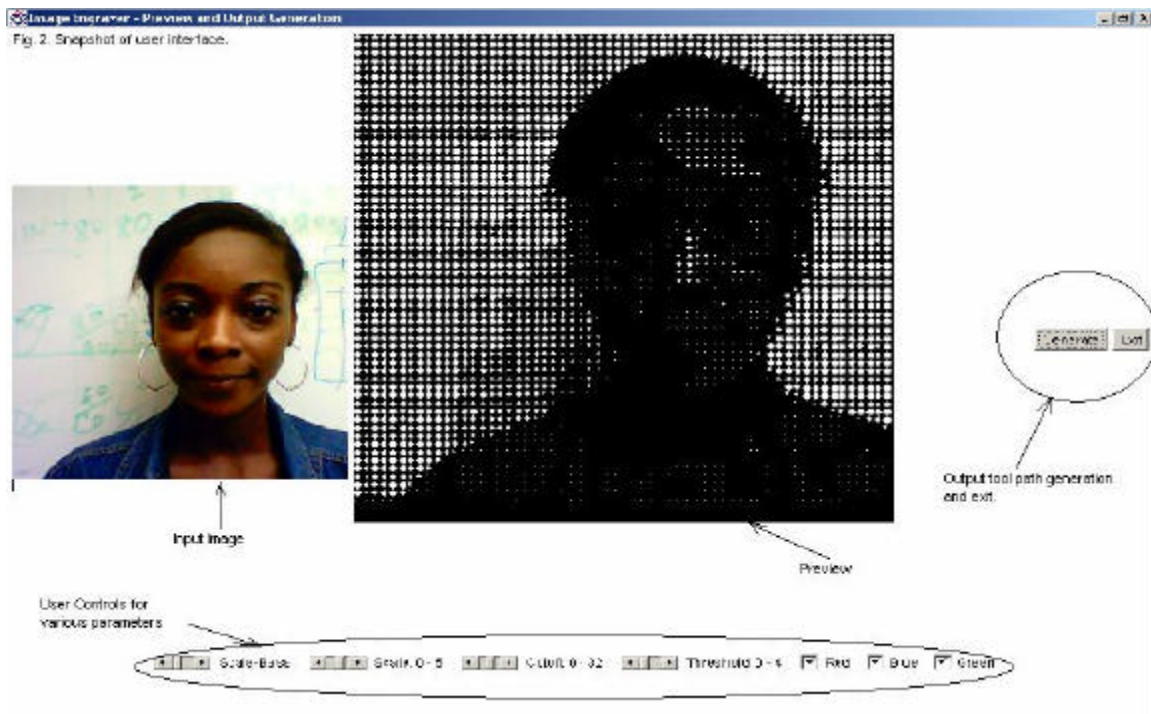
A fixed Intensity→Depth mapping for producing portraits of any person's image gave varying results. In some cases, the portrait was of very poor quality and the image was

not at all identifiable. A more common occurrence was that the surface of the foam material would be shredded. This happened because the brighter points in the image mapped onto a big diameter of the holes to be drilled. This resulted in “overlapping neighboring circles” and caused the surface to be shredded. On the other hand, some dark regions were mapped insignificantly and not seen in the portrait. Results of these were moderate to complete distortion of the final portrait. The user had no way of knowing this before generating the portrait.

This necessitated a way of generating a preview and “user controls” over parameters of the algorithm (Fig. 2).

The basic parameters for which controls were provided are:

- Scale: The portrait diameters are multiplied by a constant scale factor.
- Cutoff: Any diameter value above this value is brought down to the cutoff value.
- Threshold: Any diameter below this value is ignored.



## Sample Outputs

The collection of fabricated portraits which were generated using the Roland EGX-300 Engraving machine are shown in Fig. 3.



Fig. 3. Sample generated outputs.

Each of the outputs had been generated by choosing a suitable scaling, cutoff and threshold values such that there wasn't any shredding (as was evident from the preview) and the output was pleasing enough.

An example where the above approach failed is the case of Natasha, a person with extremely fair complexion.

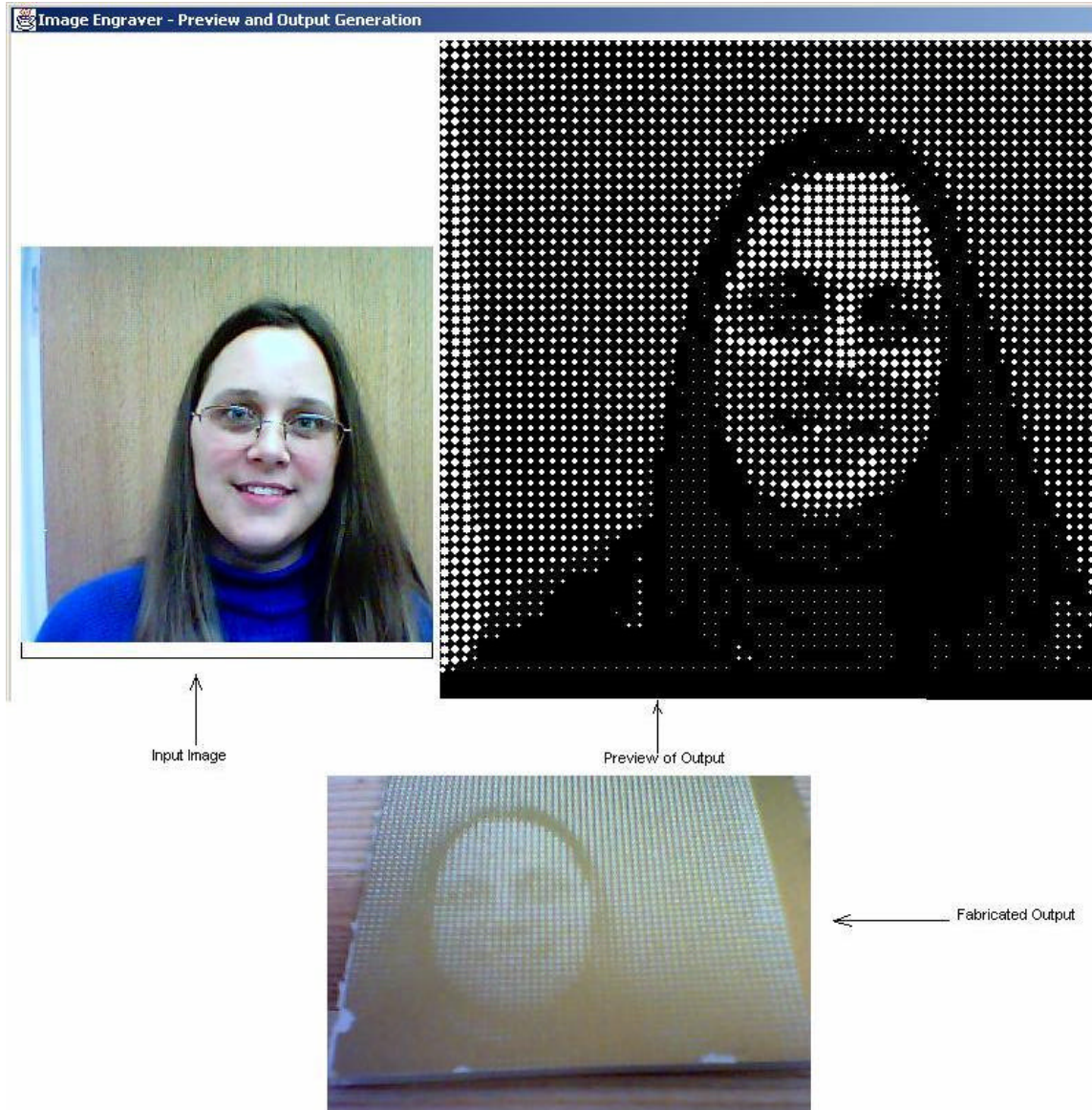


Fig. 4. Preview and output of Natasha's image.

The output produced was bland and had a “ghost-like” appearance. This could not be salvaged by adjusting the three control parameters. In order to correct this, a set of new controls were added:

#### 1) User Selection of Spectral planes

Initially, all three spectral planes were considered (by cross-averaging). In some cases this leads to a bland output like in Natasha's case (above). The idea of cross-averaging across the planes did not work in this case because spectral information across three planes was being forced onto a single surface. So, there needed to be a method to provide a way to select which of the planes need to be used.

This feature is also helpful when the image consists predominantly of one color plane with the other color planes having very insignificant values. This can be observed with the case of “kiki” [Fig. 5a]. Kiki’s predominantly brown and was photographed in yellow lighting. When cross-averaged across all the planes, kiki was almost absent as can be seen from the preview.

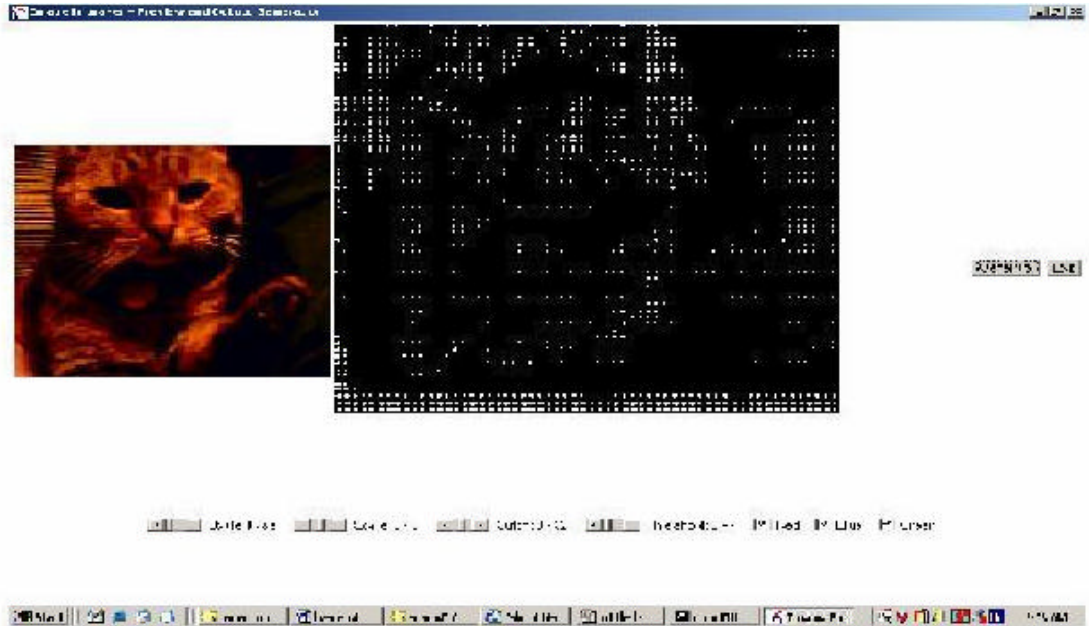


Fig. 5a. A portrait of Kiki, generated by cross-averaging across all the spectral planes. The preview shows that most of the data is lost.

An explanation for this is most of kiki's image data are concentrated in one of the spectral planes and the other spectral planes being insignificant, cross-averaging kills the output values. On, selecting the red plane alone, kiki could be identified (as seen from the preview – Fig. 5b.).

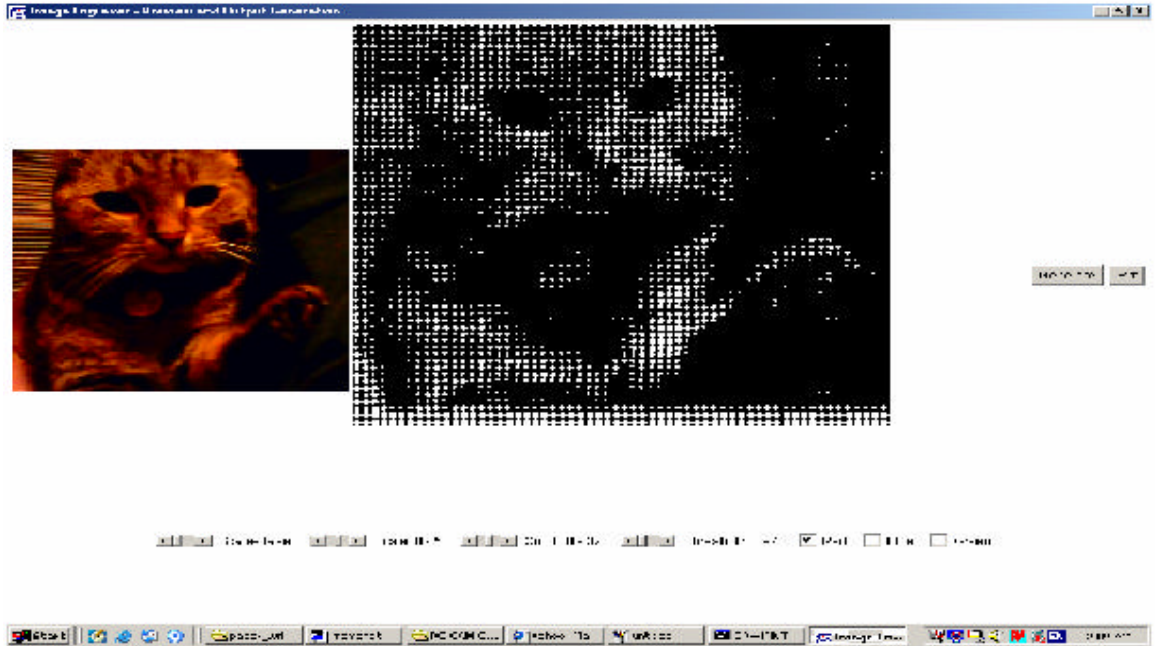


Fig. 5b) A preview of Kiki's portrait with the red plane alone selected.

Thus, a provision was made to allow the user to select which spectral plane(s) to use. By this way, only that plane (or combination of plane) which gave a contrasting output could be selected. This led to a significant difference in the contrast of the fabricated output as can be seen from the new portrait of Natasha(Fig. 6).



Fig. 6. Natasha, when generated on the "GREEN" plane alone.

As can be seen facial features are more apparent. Also, other features like the band around the neck are now evident.

## 2) Control of Intensity-Depth Mapping function \*

The mapping function generates a map from the intensity values to the depth of the drill. Initially the idea was to control the amount area exposed in proportion to the intensity value.

We can see that,

Depth of drill  $\propto$  Diameter of circle drilled

$(\text{Diameter of circle drilled})^2 \propto \text{Area of core exposed.}$

This meant that if the area exposed is to be proportional to the intensity values, the depth should be made proportional to the square root of the intensity. Thus:

Depth of drill  $\propto (\text{Intensity})^{1/2}$

→  $(\text{Intensity})^{1/2} \propto \text{Diameter of circle drilled}$

→  $[(\text{Intensity})^{1/2}]^2 \propto \text{Area exposed.}$

→  $\text{Intensity} \propto \text{Area exposed.}$

This was the function then implemented. The square root of the intensity value was taken as the depth setting for the cutting tool.

In the case of Natasha, however, a linear relationship between intensity and tool depth produced a better result (Fig. 7):

---

\* This feature has not yet been implemented as a user control and was experimented with by brute modification of the algorithm.



Fig 7. Natasha's portrait using a linear Intensity->Depth mapping function.

The linear Intensity  $\rightarrow$  Depth mapping function produced a marked change in output appearance of Natasha's portrait. This can be attributed to the different properties of the two mapping functions. When we follow the linear mapping function, the area exposed becomes

$$\text{Area exposed } \mathbf{a} [\text{Intensity}]^2$$

because intensity has been made proportion to the depth, and the area exposed is proportional to the square of the depth.

This causes the higher intensity data points to be accentuated in contrast, while those on the lower side of the intensity spectrum are rendered less significant. This improves the appearance of images of people with pale skin, but has the opposite effect on images of people with a darker complexion.

This suggests providing the user with a control over the kind of mapping function. (or with a range of mapping functions of the form,  $y = x^a$ , where the  $\mathbf{a}$ -value can be controlled by the user [Fig. 8]).

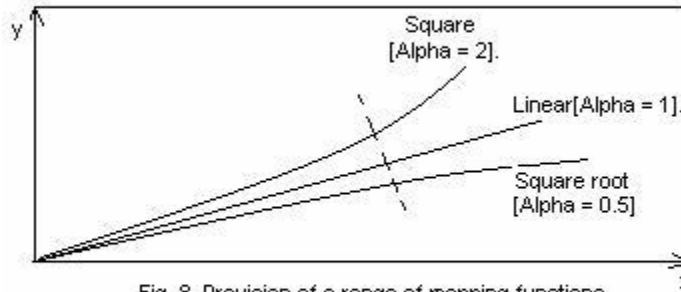


Fig. 8. Provision of a range of mapping functions  $y = f(x)$ . Alpha values vary from 0.5 to 2.

### 3) Base-Scalability [Fig. 9]:

Instead of scaling all the diameter values in the portrait simultaneously, a tool was provided to only selectively scale those points which had a diameter above a particular value (referred to as scale-base value). If the scale base is set to 0 (default state), then the base-scalability acts just like the normal scaling function.

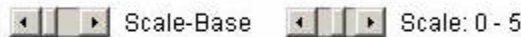


Fig. 9. Base-Scalability implemented using a pair of scroll bars. One selects the base and the other determines the scaling factor.

The base-scalability feature allows a selective scaling of specific regions which are brighter. The motivation is similar to that for selecting the mapping function, where certain regions (of a specific brightness range) can be made more prominent than the others. With a simple implementation for selection of the mapping function however, the base-scalability feature should be a redundant one and need not be provided.

### Implementation Details

A control class is maintained. This class basically maintains various flags and variables which are directly modified depending on the state of the user control objects like sliders and check boxes.

```

public class Control
{
    public int scale = 5;           //Determines scaling factor
    public int cutoff = 16;        //Determines cutoff parameter
    public int threshold = 5;      //Determines threshold level
    public int red = 1;            // A flag indicating whether to include red plane.
    public int green = 1;          // A flag indicating whether to include green plane.
    public int blue = 1;           // A flag indicating whether to include blue plane.
    public int scalebase = 0;      // Scalebase value is determined by this.

    public void Control( );        // No argument constructor

    public void Control(int sc, int cut, int thres, int red, int green, int blue, int
sclaebase)                        //Constructor with initializing arguments.
}

```

A control class object is first initialized at the start of the application. The user modifies the contents of this object by interacting with the controls provided in the form of scroll bars, checkboxes and buttons. Every time such an event occurs (like the movement of the slider bar), the control object is modified. The tool path generating function and the preview functions refer to this control object to determine the tool path and the preview respectively.

The complete java code is posted online and can be viewed at the following link:

[www.cs.uml.edu/~fredm/courses/91.548/student/kramanat/projcode](http://www.cs.uml.edu/~fredm/courses/91.548/student/kramanat/projcode)

### **Other aspects of the application**

The other aspects of the application include generation of a tool path on the Roland EGX-300 Engraver. On user's request, the application generates a file containing instructions in CAMM-GL I. These, are fed to the Roland Engraver to produce the fabricated output. The time of fabrication was reduced by minimizing inter-drill travel time of the pen.

### **Conclusion**

An algorithm for generating engraved portrait on a surface of coated foam was developed. The algorithm was tested on a number of test inputs (images). Based on the results of these, a number of control parameters were added to the algorithm which could be varied by the user. Using these set of parameters, the quality of the fabricated portrait was controlled.

A large number of controls may be added. These may not necessarily improve the utility. For example, it seems that the base-scalability feature can be removed by providing a

method to control the Intensity→Depth mapping function. At the same time, more controls may result in inconvenience for the user. For example, if there were a half a dozen scroll bars and half a dozen checkboxes for controlling different parameters, naïve users may find the interface inconvenient. There is a good chance they would will be lost using the tools which were supposed to guide them.

By exploring the various possibilities in user control, a minimal set of controls need to be arrived at which can generate portraits of different images. The direction seems to be to have a few set of controls for modifying mapping function rather than having a fixed mapping function and many controls to modify the mapped output. There is definitely scope in improving the utility aspect of this application and then enhancing its usability for different types of users from children to advanced users (who may be provided with more technical detail).

## References

1. Roland EGX-300 Desktop Engraver User's Manual
2. [java.sun.com/j2se/1.3/docs/api/](http://java.sun.com/j2se/1.3/docs/api/)
3. [acomp.stanford.edu/acpubs/Docs/graphic\\_file\\_formats/](http://acomp.stanford.edu/acpubs/Docs/graphic_file_formats/)